Puerto Rican boa PVA

Anna Tucker

4/23/2020

This document contains the code to run the projection model for Puerto Rican boas described in Tucker, A.M., C.P. McGowan, E. Mulero, N.F. Angeli, and J.P. Zegarra. 2020. A demographic projection model to support conservation decision making for an endangered snake with limited monitoring data.

All other code, including sensitivity analysis, is available online at https://www.github.com/annamtucker/puerto-rican-boa.

Contact Anna Tucker (annamtucker@gmail.com) with questions.

Functions

First we define the functions used in the population projection.

Beta distribution functions using methods of moments calculation.

```
beta_shape_parms = function(mean, sd){
  avg = mean
  var = sd*sd

  #a = ifelse(var < avg*(1-avg), avg*(((avg*(1-avg))/var)-1), 100*avg)
  #b = ifelse(var < avg*(1-avg), (1-avg)*(((avg*(1-avg))/var)-1), 100*(1-avg))

  a = 100*avg
  b = 100*(1-avg)

  return(list(a = a, b = b))
}

get_beta_vals = function(n, mean, sd){
  x = beta_shape_parms(mean, sd)
  rbeta(n, x$a, x$b)
}
```

Calculate lambda from projection outputs and find geometric mean.

```
calc_lambda = function(x){
  lam = x$N[2:length(x$N)]/x$N[1:(length(x$N)-1)]
  return(c(lam, NA))
}

geo_mean = function(lambda, N){
  if(length(which(N == 0)) > 0){
    ext = min(which(is.na(lambda)))
    y = lambda[1:(ext-2)]
  } else y = lambda[1:(length(lambda)-1)]
```

```
  n = length(y)
  z = prod(y, na.rm = T)
  z^(1/n)
}
```

This function draws parameter values from user-defined distributions.

```
draw_parms = function(iter, N = TRUE){

  # survival and transition rates
  surv = avg.mat*surv.mask
  surv.sd = para.sd.mat*surv.mask

  avg.surv = matrix(get_beta_vals(16, surv, surv.sd),
                    nrow = 4, ncol = 4)


  # fecundity
  fec = avg.mat*fec.mask
  fec.sd = para.sd.mat*fec.mask

  avg.fec = matrix(rlnorm(16, log(fec), fec.sd),
                   nrow = 4, ncol = 4)

  # output matrices
  avg = avg.surv + avg.fec
  if(N) Ninit = round(runif(1, Ninit.min, Ninit.max))
  K = round(runif(1, K.min, K.max))

  if(N) {
    ret <- tibble(avg = list(avg),
                  Ninit = Ninit,
                  K = K)
  } else ret <- tibble(avg = list(avg),
                       K = K)

  return(ret)

}
```

This function projects the population for a given matrix of demographic rates (`avg`), proportion of habitat in urban areas (`prop.hab`), rate of urbanization (`rate`), initial population size (`Ninit`) and carrying capacity (`K`).

```
project_pop = function(avg, prop.hab, rate, Ninit, K){

  realK = numeric(30)
  realK[1] <- K

  for(t in 2:30){
    realK[t] <- round(realK[t-1] - rate*realK[t-1])
  }


  ### natural habitat
```

```r
# temporal variation
surv = avg*surv.mask

# to make sure all survivals < 1 (urban)
if(length(which(surv > 1)) > 0){
  surv[which(surv > 1)] <- 1
}
surv.sd = surv*temp.cv

fec = avg*fec.mask
fec.sd = fec*temp.cv
if(fec.sd[1,3] < 1){fec.sd[1,3] = 9*0.15}
if(fec.sd[1,4] < 1){fec.sd[1,4] = 9*0.15}

### urban habitat
avg.U <- runif(16, urban[1], urban[2])*avg

# temporal variation
surv.U = avg.U*surv.mask

# to make sure all survivals < 1 (urban)
if(length(which(surv.U > 1)) > 0){
  surv.U[which(surv.U > 1)] <- 1
}
surv.sd.U = surv.U*temp.cv

fec.U = avg.U*fec.mask
fec.sd.U = fec.U*temp.cv
if(fec.sd.U[1,3] < 1){fec.sd.U[1,3] = 9*0.15}
if(fec.sd.U[1,4] < 1){fec.sd.U[1,4] = 9*0.15}

# 1 = natural, 2 = urban
mat = array(NA, dim = c(2, n.years, 4, 4))
for(i in 1:n.years){
  yr.surv = matrix(get_beta_vals(16, surv, surv.sd),
                   nrow = 4, ncol = 4)
  yr.fec = matrix(rlnorm(16, log(fec), log(fec.sd)),
                  nrow = 4, ncol = 4)
  if(length(which(yr.fec > max.fec)) > 0) {
    yr.fec[which(yr.fec > max.fec)] <- max.fec
  }
  yr.fec[which(is.na(yr.fec))] <- 0
  mat[1,i,,] <- yr.surv+yr.fec


  yr.surv.U = matrix(get_beta_vals(16, surv.U, surv.sd.U),
                     nrow = 4, ncol = 4)
  yr.fec.U = matrix(rlnorm(16, log(fec.U), log(fec.sd.U)),
                    nrow = 4, ncol = 4)
  if(length(which(yr.fec.U > max.fec)) > 0) {
    yr.fec.U[which(yr.fec.U > max.fec)] <- max.fec
  }
  yr.fec.U[which(is.na(yr.fec.U))] <- 0
```

```
    mat[2,i,,] <- yr.surv.U+yr.fec.U
  }

  # start population at stable stage distribution from average matrix
  dist = eigen.analysis(avg)$stable.stage
  dist.U = eigen.analysis(avg.U)$stable.stage

  # population projection matrix
  N = array(NA, dim = c(2, 4, n.years))
  N[1,,1] <- round((Ninit*(1-prop.hab))*dist)
  N[2,,1] <- round((Ninit*prop.hab)*dist.U)


  # project with temporal variation
  for(i in 2:n.years){

    # set fecundity = 0 if pop exceeds K
    if(sum(N[,,i-1]) > realK[i]){
      mat[,i-1,1,3:4] <- 0
    }

    N[1,,i] <- round(mat[1,i-1,,] %*% N[1,,i-1])
    N[2,,i] <- round(mat[2,i-1,,] %*% N[2,,i-1])

    switch <- round(rate*N[1,,i])

    N[1,,i] <- N[1,,i]-switch
    N[2,,i] <- N[2,,i]+switch
  }

  Ntot = apply(N, 3, sum) # total pop size
  lambda = c(Ntot[2:n.years]/Ntot[1:(n.years-1)], NA)

  out <- tibble(Ntot = Ntot,
                realK = realK,
                year = c(1:n.years),
                lambda = lambda)

  return(out)
}
```

Define input parameters

```
set.seed(419)

# simulation parameters
n.iters = 1000                # for parametric uncertainty (10000 reps used in final version)
n.reps = 1                    # replicates of each parameter draw
n.years = 30                  # number of years to project
n.habs = 2

# scenarios
prop.urban = 0.43
```

```
habitat.area = 379029
urb.rates = c(0, 0.008, 0.016, 0.024)  # rate of urban growth per year


# demographic parameters

# initial population size
Ninit.min = habitat.area*0.1
Ninit.max =  habitat.area*0.5

# maximum population size
K.min = habitat.area*1
K.max = habitat.area*3

# demographic rate CVs
para.cv = 0.15
temp.cv = 0.15

# adjustment on average demographic rates for urban habitats
urban = c(0.5, 1)

# survival rates
young.surv = 0.3
juv.surv = 0.9
subad.surv = 0.72
ad.surv = 0.9

# growth rates
yj = 0.67
js = 0.55
sa = 0.25

# fecundity and breeding propensity of subadults
f = 4.5
subad.bp = 2/f
max.fec = 35

# matrix parameters
T.YY = young.surv * (1-yj)
T.JY = young.surv * yj
T.JJ = juv.surv * (1-js)
T.SJ = juv.surv * js
T.SS = subad.surv * (1-sa)
T.AS = subad.surv * sa
T.AA = ad.surv
F.SY = f * subad.bp * young.surv
F.AY = f * young.surv

avg.mat = matrix(c(T.YY, 0, F.SY, F.AY,
                   T.JY, T.JJ, 0, 0,
                   0, T.SJ, T.SS, 0,
                   0, 0, T.AS, T.AA),
                 nrow = 4, ncol = 4, byrow = T)
```

```
# parametric uncertainty SD
para.sd.mat = avg.mat*para.cv

# manually change sd for subad fecundity (less than 1 otherwise and leads to error)
#para.sd.mat[1,c(3,4)] <- c(log(F.SY)*0.15, log(F.AY)*0.15)

# to isolate survival rates only
surv.mask = matrix(c(1, 0, 0, 0,
                     1, 1, 0, 0,
                     0, 1, 1, 0,
                     0, 0, 1, 1),
                   nrow = 4, ncol = 4, byrow = T)

# to isolate fecundities only
fec.mask = matrix(c(0, 0, 1, 1,
                    0, 0, 0, 0,
                    0, 0, 0, 0,
                    0, 0, 0, 0),
                  nrow = 4, ncol = 4, byrow = T)
```

Run simulation

Running the projection is simplified using functions and the `map` functions.

```
iters <- as_tibble(expand.grid(rate = urb.rates,
                               prop.hab = prop.urban,
                               iter = c(1:n.iters))) %>%
  mutate(parms = purrr::map(iter, draw_parms)) %>%
  unnest()

sim <- iters %>%
  mutate(projection = pmap(list(avg, prop.hab, rate, Ninit, K), project_pop))

Ntot <- sim %>%
  select(iter, rate, projection, Ninit, K) %>%
  unnest()
```

Results

Change in population size over time

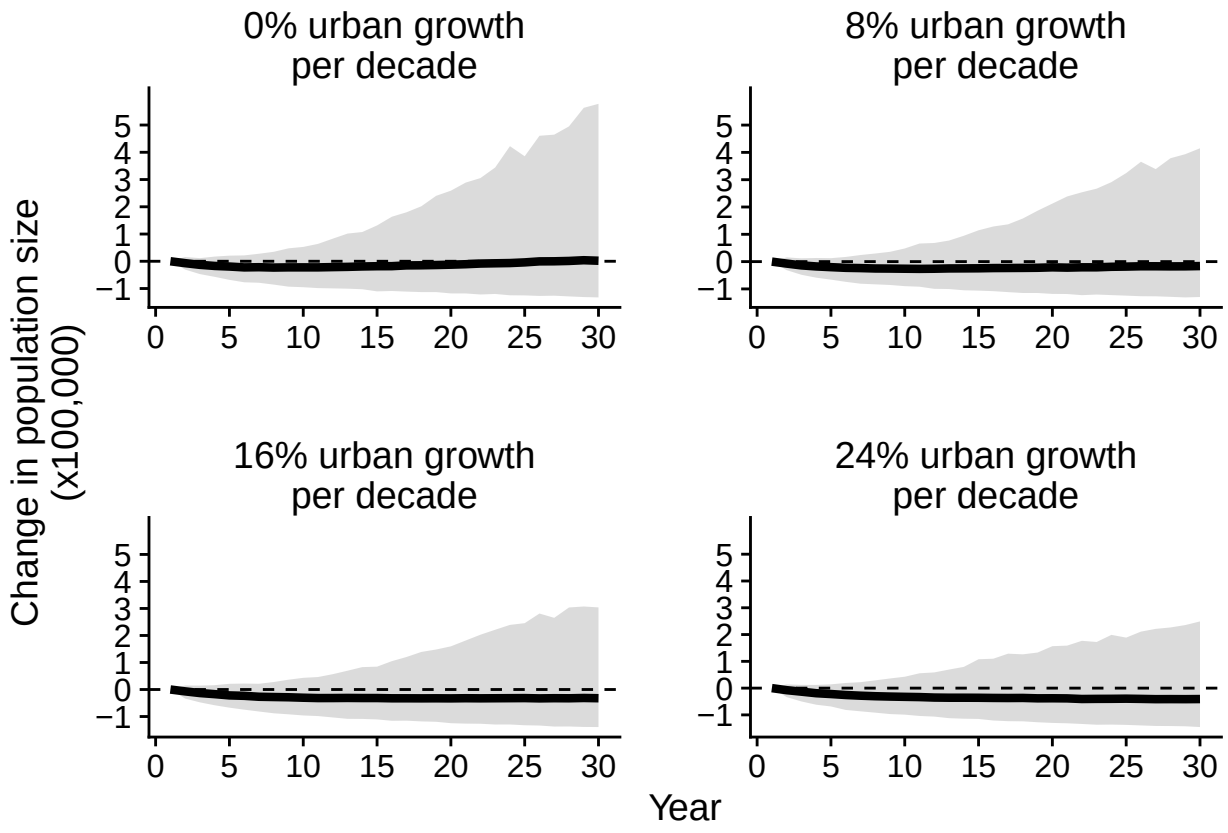
```
percent = gsub(" ", "", paste(urb.rates*1000, "%"))

scenario.labs = sapply(percent, FUN = function(x) paste(x, "urban growth\nper decade"))

labels <- tibble(rate = urb.rates,
                 scenario = scenario.labs) %>%
  mutate(scenario = fct_reorder(scenario, rate))

# population size over time
Nplot <- Ntot %>%
  full_join(labels) %>%
  group_by(scenario, iter, year, Ninit) %>%
  summarize(Ntot = sum(Ntot)) %>%
```

```
  ungroup() %>%
  mutate(deltaN = Ntot-Ninit) %>%
  group_by(scenario, year) %>%
  summarize(medN = median(deltaN, na.rm = T),
            lciN = quantile(deltaN, 0.025, na.rm = T),
            uciN = quantile(deltaN, 0.975, na.rm = T)) %>%
  ungroup() %>%
  ggplot(aes(x = year, y = medN/100000)) +
  geom_ribbon(aes(ymin = lciN/100000, ymax = uciN/100000), fill = "gray80", alpha = 0.7) +
  geom_line(lwd = 1.5) +
  geom_hline(yintercept = 0, lty = 2) +
  ylab("Change in population size\n(x100,000)") +
  xlab("Year") +
  scale_x_continuous(breaks = seq(0, 30, 5), limits = c(1, 30)) +
  facet_wrap(~scenario, scales = "free") +
  scale_y_continuous(limits = c(NA, 6), breaks = c(-1, 0, 1, 2, 3, 4, 5)) +
  theme(strip.background = element_rect(fill = "white"),
        strip.text = element_text(size = 14, margin = margin(2,2,2,2)),
        panel.spacing = unit(2, "lines"))
Nplot
```



Probabilities of quasi-extinction, growth, or decline.

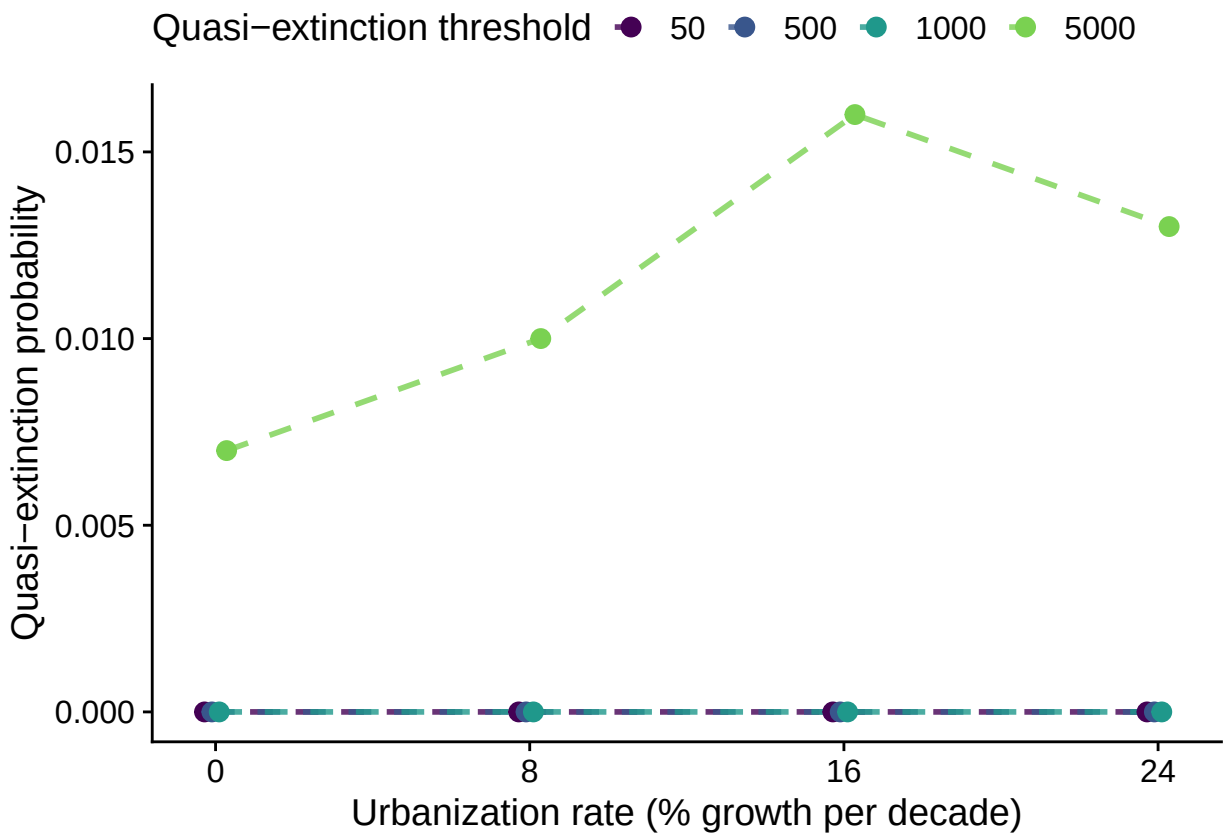
```
probs <- Ntot %>%
  rename(scenario = rate) %>%
  group_by(scenario, iter, year) %>%
```

```
  summarize(N = sum(Ntot)) %>%
  ungroup() %>%
  group_by(scenario, iter) %>%
  nest() %>%
  mutate(lambda = map(data, calc_lambda)) %>%
  unnest() %>%
  group_by(scenario, iter) %>%
  summarize(inc = ifelse(geo_mean(lambda, N) >= 1, 1, 0),
            dec = ifelse(geo_mean(lambda, N) < 1, 1, 0),
            qe_50 = ifelse(any(N < 50), 1, 0),
            qe_500 = ifelse(any(N < 500), 1, 0),
            qe_1000 = ifelse(any(N < 1000), 1, 0),
            qe_5000 = ifelse(any(N < 5000), 1, 0)) %>%
  ungroup() %>%
  #mutate(Scenario = scenario.labs[scenario]) %>%
  group_by(scenario) %>%
  summarize(pext_50 = mean(qe_50),
            pext_500 = mean(qe_500),
            pext_1000 = mean(qe_1000),
            pext_5000 = mean(qe_5000),
            pinc = mean(inc),
            pdec = mean(dec))
probs
```

```
## # A tibble: 4 x 7
##   scenario pext_50 pext_500 pext_1000 pext_5000  pinc  pdec
##      <dbl>   <dbl>    <dbl>     <dbl>     <dbl> <dbl> <dbl>
## 1    0           0        0         0     0.007 0.51  0.49
## 2    0.008       0        0         0     0.01  0.43  0.570
## 3    0.016       0        0         0     0.016 0.357 0.643
## 4    0.024       0        0         0     0.013 0.282 0.718
```

Plot quasi-exinction probability.

```
qe_plot <- probs %>%
  gather(prob, val, 2:5) %>%
  mutate(prob = fct_relevel(prob, "pext_50", "pext_500", "pext_1000", "pext_5000")) %>%
  ggplot(aes(x = scenario*1000, y = val, color = prob)) +
  geom_path(lty = 2, lwd = 1, position = position_dodge(width = 0.75), alpha = 0.8) +
  geom_point(size = 3, position = position_dodge(width = 0.75)) +
  xlab("Urbanization rate (% growth per decade)") +
  ylab("Quasi-extinction probability") +
  scale_x_continuous(breaks = c(0, 8, 16, 24)) +
  scale_color_viridis_d(name = "Quasi-extinction threshold",
                        labels = c(50, 500, 1000, 5000),
                        option = "D",
                        end = 0.8) +
  theme(legend.position = "top")
qe_plot
```


Quasi−extinction threshold
50
500
1000
5000
Quasi−extinction probability
0.015
0.010
0.005
0.000
0
8
16
24
Urbanization rate (% growth per decade)